

0. Escenario

El protocolo DHCP permite asignar y configurar automáticamente direcciones IPs en los equipos de una red. Generalmente no es importante conocer exactamente cuál es la dirección que tenemos asignada pero en ocasiones necesitamos contactar directamente con un equipo por lo que se hace necesario que la asignación de IP sea conocida y preferentemente fija. El protocolo DHCP permite estas asignaciones fijas pero supone añadir un nuevo punto de fallo haciendo que necesitemos otro servidor con el servicio de DHCP en la red.

En la vida real, para el uso diario de nuestro teléfono móvil no necesitamos conocer nuestro número de teléfono y probablemente el de nadie puesto que los tenemos en la aplicación de contactos. En ocasiones tendremos que conocer nuestro número de teléfono para que otros puedan contactar con nosotros y además sería ideal que nuestro número de teléfono no cambiara y fuera siempre el mismo.

Por defecto, la mayoría de los sistemas operativos vienen configurados para escuchar y hacer peticiones al protocolo DHCP de manera que se le asigne una dirección IP automáticamente.

Independientemente de si usamos DHCP o una configuración manual necesitaremos 4 datos:

- La dirección IP. Será nuestro identificador único en la red.
- La máscara. Sirve para identificar y separar la parte de red de la parte de host en una IP
- La puerta de enlace. Indica la IP del equipo que da salida al exterior de la red local.
- Los servidores DNS. Indica las IPs de los equipos que pueden resolver nombres e IPs

Si tienes un sistema operativo con interfaz gráfica tendrás que buscar el icono de redes y cambiar el modo automático a manual y configurar esos 4 datos donde corresponda.

Todos los sistemas operativos tienen una interfaz CLI donde podemos interactuar con el sistema a base de comandos. Esta manera de interactuar es menos amigable que usar la interfaz gráfica pero mucho más potente.

En esta guía configuraremos un equipo con una IP fija y comprobaremos que funciona correctamente.

1. Comprobar el estado actual.

Al iniciar sesión en un Ubuntu Server ya aparecen algunos datos interesantes de la configuración de red.

```
Slave_01
Ubuntu 22.04.3 LTS ubuntuserver tty1
ubuntuserver login: administrador
Password:
Welcome to Ubuntu 22.04.3 LTS (GNU/Linux 5.15.0-79-generic aarch64)

* Documentation:  https://help.ubuntu.com
* Management:    https://landscape.canonical.com
* Support:        https://ubuntu.com/advantage

System information as of sáb 26 ago 2023 10:50:33 UTC

System load:                0.0
Usage of /:                  11.6% of 47.41GB
Memory usage:               9%
Swap usage:                 0%
Processes:                  99
Users logged in:            0
IPv4 address for enp0s1: 10.0.0.3
IPv6 address for enp0s1: fe4:6a6e:e9ed:5387:893:bbff:fe97:33b

* Strictly confined Kubernetes makes edge and IoT secure. Learn how MicroK8s
  just raised the bar for easy, resilient and secure K8s cluster deployment.

  https://ubuntu.com/engage/secure-kubernetes-at-the-edge

El mantenimiento de seguridad expandido para Applications está desactivado
Se pueden aplicar 0 actualizaciones de forma inmediata.

Active ESM Apps para recibir futuras actualizaciones de seguridad adicionales.
Vea https://ubuntu.com/esm o ejecute «sudo pro status»

Last login: Fri Aug 25 21:31:00 UTC 2023 on tty1
administrador@ubuntuserver:~$
```

En esta captura podemos ver que hay una interfaz física de red llamada **enp0s1** y que tiene asignada la dirección IP **10.0.0.3**.

Con el siguiente comando (ip) y el parámetro adecuado (address) podemos ver información extendida.

```
ip address
```

En muchas ocasiones podemos los parámetros muy abreviados de manera que "address" quede solo en una "a". El resultado del comando lo podemos ver en la siguiente captura:

```

Slave_01
administrador@ubuntuserver:~$ ip address
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
        valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host
        valid_lft forever preferred_lft forever
2: enp0s1: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group default qlen 1000
    link/ether 0a:93:bb:97:03:3b brd ff:ff:ff:ff:ff:ff
    inet 10.0.0.3/24 metric 100 brd 10.0.0.255 scope global dynamic enp0s1
        valid_lft 86370sec preferred_lft 86370sec
    inet6 fd84:6a8e:e3ed:5387:893:bbff:fe97:33b/64 scope global dynamic mngtmpaddr noprefixroute
        valid_lft 2591989sec preferred_lft 604789sec
    inet6 fe80::893:bbff:fe97:33b/64 scope link
        valid_lft forever preferred_lft forever
administrador@ubuntuserver:~$

```

Lo que vemos aquí es que el sistema operativo detecta 2 interfaces. La primera es la interfaz "lo" o loop con la ip 127.0.0.1 que siempre apunta al propio equipo. Es la manera sencilla de hacerse referencia a uno mismo cuando el propio equipo es a la vez cliente y servidor.

La interfaz que nos interesa es la 2 con el nombre "enp0s1". En versiones anteriores los nombres de las interfaces de red eran más sencillos como por ejemplo "eth0", "eth1", "wlan0", "wlan1". El problema es que no estaba garantizado que una misma interfaz siempre se llamara de la misma manera especialmente cuando se añadía o eliminaba hardware. Ahora los nombres de las interfaces de red son más complicados pero también más descriptivos ya que informan de la interfaz y de la posición en la ranura PCI Express que ocupa. Dicho esto tened en cuenta que vuestra pantalla puede tener datos distintos porque tenéis la tarjeta de red pinchada en otra ranura o tal vez estéis usando un adaptador wifi.

Como curiosidad el nombre "enp0s1" hay que interpretarlo como "en" indicando que es una interfaz ethernet, "p0" que indica la localización geográfica en PCI y "s1" que indica el slot donde está pinchada la tarjeta.

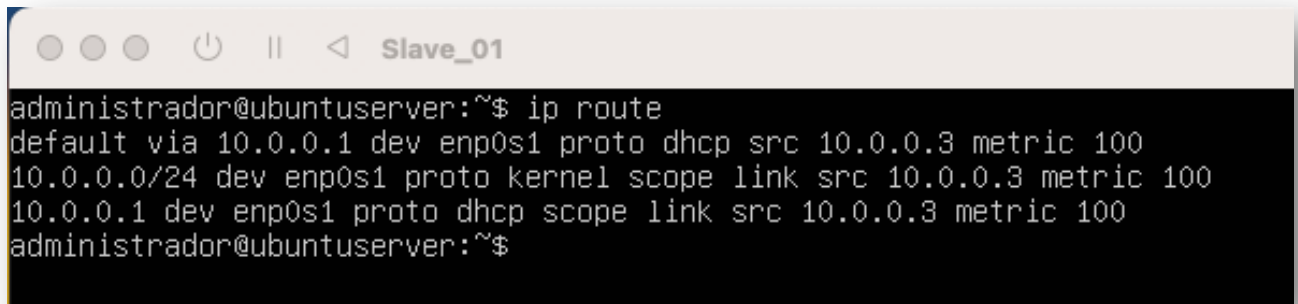
Fijaros en la captura anterior para localizar la siguiente información:

- La IP asignada es la 10.0.0.3
- La máscara usa 24 unos a la izquierda, o lo que es lo mismo, 255.255.255.0
- La IP ha sido asignada dinámicamente, es decir a través de un DHCP

Solo nos queda averiguar la IP del equipo que sirve de puerta de enlace. Para ello usaremos el mismo comando con distinto parámetro:

```
ip route
```

Con este comando obtenemos las rutas que conoce el sistema para alcanzar distintos destinos y de esta manera podemos averiguar la IP que sirve de puerta de enlace o Gateway.



```
administrador@ubuntuserver:~$ ip route
default via 10.0.0.1 dev enp0s1 proto dhcp src 10.0.0.3 metric 100
10.0.0.0/24 dev enp0s1 proto kernel scope link src 10.0.0.3 metric 100
10.0.0.1 dev enp0s1 proto dhcp scope link src 10.0.0.3 metric 100
administrador@ubuntuserver:~$
```

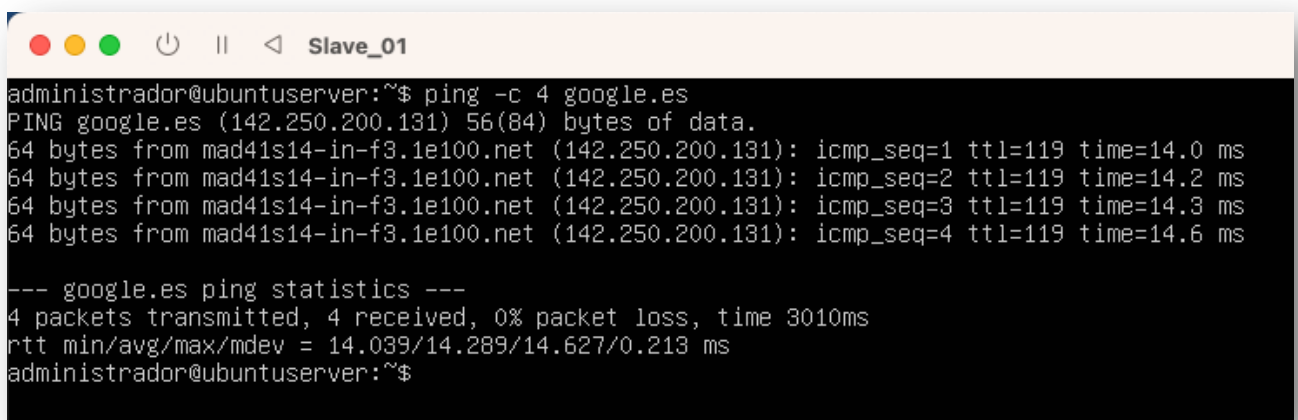
En este equipo solo hay una interfaz de red por lo que es sencillo imaginar que todo saldrá por esta única interfaz. Analizando la salida del comando obtenemos la siguiente información:

- Para llegar al equipo 10.0.0.1 usaremos la interfaz enp0s1
- Para llegar a cualquier equipo de la red 10.0.0.x usaremos la interfaz enp0s1
- Si no conocemos la manera de llegar usaremos la ruta por defecto hacia el equipo 10.0.0.1

Por último podemos comprobar que tenemos conectividad con el siguiente comando:

```
ping -c 4 google.es
```

Este comando se limita a enviar 4 paquetes a un destino, Google.es en este caso, y esperar a que “reboten” y vuelvan. La siguiente es la salida que indica que todo ha ido bien.



```
administrador@ubuntuserver:~$ ping -c 4 google.es
PING google.es (142.250.200.131) 56(84) bytes of data:
64 bytes from mad41s14-in-f3.1e100.net (142.250.200.131): icmp_seq=1 ttl=119 time=14.0 ms
64 bytes from mad41s14-in-f3.1e100.net (142.250.200.131): icmp_seq=2 ttl=119 time=14.2 ms
64 bytes from mad41s14-in-f3.1e100.net (142.250.200.131): icmp_seq=3 ttl=119 time=14.3 ms
64 bytes from mad41s14-in-f3.1e100.net (142.250.200.131): icmp_seq=4 ttl=119 time=14.6 ms

--- google.es ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3010ms
rtt min/avg/max/mdev = 14.039/14.289/14.627/0.213 ms
administrador@ubuntuserver:~$
```

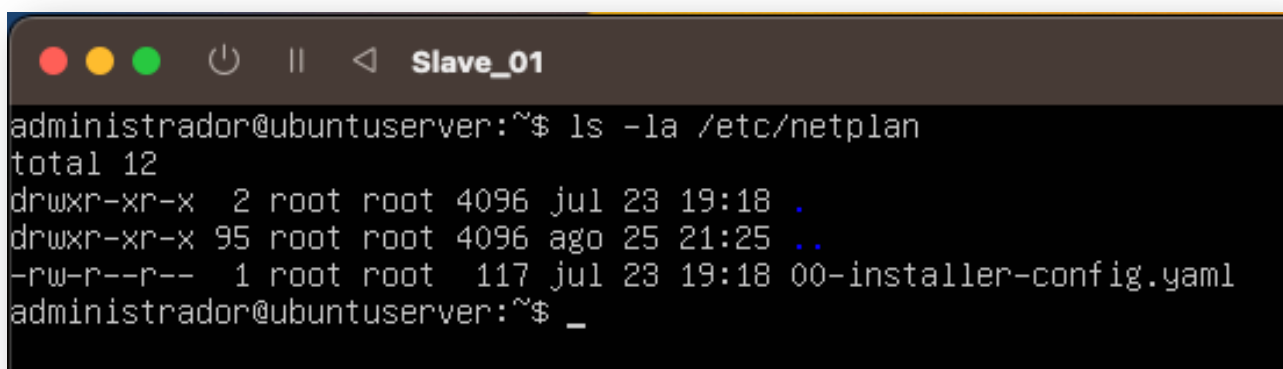
2. Cambios persistentes

Podemos cambiar la IP de un equipo manualmente pero estos cambios son efímeros y en el siguiente encendido volveremos a cargar la configuración por defecto. En Linux generalmente guardamos los archivos de configuración en la carpeta **/etc/**.

En las versiones más actuales de Ubuntu y cada vez más distribuciones Linux usamos **netplan** como herramienta gestora de red. Encontraremos el archivo de configuración en la ruta **/etc/netplan**. En este caso el archivo es de tipo **YAML**. Este formato prioriza la legibilidad de manera que una persona pueda interpretar fácilmente un archivo con un simple editor de texto. La sintaxis es sencilla y consiste en pares de clave:valor donde la indentación es importante así que debemos prestar atención a los espacios en blanco que pudiera haber delante, son importantes.

```
ls -la /etc/netplan
```

Antes de editar el archivo tenemos que averiguar su nombre. El comando "ls" junto con una ruta nos lista los archivos y carpetas que hay allí. En este caso, los modificadores "-la" sirven para mostrar información en forma de lista y todos los archivos, incluso los ocultos.



```
administrador@ubuntuserver:~$ ls -la /etc/netplan
total 12
drwxr-xr-x  2 root root 4096 jul 23 19:18 .
drwxr-xr-x 95 root root 4096 ago 25 21:25 ..
-rw-r--r--  1 root root  117 jul 23 19:18 00-installer-config.yaml
administrador@ubuntuserver:~$ _
```

En la captura anterior se muestra la salida del comando y podemos ver que el archivo que buscamos se llama **"00-installer-config.yaml"**. Realmente el nombre no es importante y en caso de conflicto se usan los primeros número para priorizar el archivo de configuración que se usará.

```
sudo nano /etc/netplan/00-installer-config.yaml
```

El comando anterior ejecuta con permisos administrativos (sudo) un editor de texto en terminal (nano) que usaremos para modificar el archivo yaml.

```
Slave_01
GNU nano 6.2
# This is the network config written by 'subiquity'
network:
  version: 2
  ethernets:
    enp0s1:
      dhcp4: false
      address:
        - 10.0.0.100/24
      nameservers:
        - 8.8.8.8
        - 10.0.0.1
      routes:
        - to: default
          via: 10.0.0.1
```

Debemos modificar el archivo para que quede así. Fijaros que las indentaciones nos permiten ver a simple vista que las IPs 8.8.8.8 y 10.0.0.1 son una lista que corresponden a servidores DNS (nameservers).

También se puede ver fácilmente que la IP 10.0.0.1 es la ruta a tomar por defecto.

Guardamos los cambios con la combinación de teclas Ctrl+X (salir) y al detectar cambios nos preguntará si queremos guardar y con qué nombre.

Una vez tenemos el archivo de configuración guardado debemos probarlo para asegurarnos que es correcto, al menos sintácticamente. Para ello usaremos el siguiente comando:

```
sudo netplan try
```

En caso de error nos advierte del problema como en la siguiente captura:

```
Slave_01
administrador@ubuntuserver:~$ netplan try
/etc/netplan/00-installer-config.yaml:7:7: Error in network definition: unknown key 'address'
  address:
  ^
administrador@ubuntuserver:~$ _
```

Si nos fijamos en el mensaje de salida que ofrece el comando veremos que la propiedad "address" no es reconocida. El nombre correcto de la clave es "addresses" porque una interfaz puede tener varias IPs asignadas simultáneamente.

Editamos el archivo de configuración como hicimos anteriormente y vamos resolviendo los problemas que pueda haber.

En <https://netplan.readthedocs.io/en/stable/> podéis encontrar la documentación oficial de netplan que os permitirá encontrar los errores y solucionarlos así como ampliarlo para cualquier montaje de red que se pueda necesitar.

En algún momento tendremos el archivo de configuración correctamente escrito y volveremos a ejecutar el comando sin errores:

```
sudo netplan try
```

Si todo va bien, netplan probará la nueva configuración durante unos segundos.

Si estás trabajando en local puedes arreglar casi todo con en terminal pero si estás conectado en remoto vas a perder conexión seguro. Piensa que si te has equivocado al poner la IP o cualquier otro error semántico no podrás volver a conectar con ese equipo.



```
Slave_01
administrador@ubuntuserver:~$ sudo netplan try
Do you want to keep these settings?

Press ENTER before the timeout to accept the new configuration

Changes will revert in 117 seconds
```

Parece una buena idea probar conectividad durante estos segundos.




```
Slave_01
administrador@ubuntuserver:~$ sudo netplan try
Do you want to keep these settings?

Press ENTER before the timeout to accept the new configuration

Changes will revert in 1 seconds
Reverting.
administrador@ubuntuserver:~$ _
```




```
~ - -zsh - 80x27
Last login: Sun Aug 27 21:22:55 on ttys000
javi@macmini ~ % ping 10.0.0.100
PING 10.0.0.100 (10.0.0.100): 56 data bytes
64 bytes from 10.0.0.100: icmp_seq=0 ttl=64 time=0.888 ms
64 bytes from 10.0.0.100: icmp_seq=1 ttl=64 time=1.007 ms
64 bytes from 10.0.0.100: icmp_seq=2 ttl=64 time=1.105 ms
64 bytes from 10.0.0.100: icmp_seq=3 ttl=64 time=1.018 ms
64 bytes from 10.0.0.100: icmp_seq=4 ttl=64 time=1.050 ms
64 bytes from 10.0.0.100: icmp_seq=5 ttl=64 time=1.113 ms
64 bytes from 10.0.0.100: icmp_seq=6 ttl=64 time=1.037 ms
64 bytes from 10.0.0.100: icmp_seq=7 ttl=64 time=0.992 ms
64 bytes from 10.0.0.100: icmp_seq=8 ttl=64 time=1.035 ms
64 bytes from 10.0.0.100: icmp_seq=9 ttl=64 time=0.862 ms
64 bytes from 10.0.0.100: icmp_seq=10 ttl=64 time=1.011 ms
64 bytes from 10.0.0.100: icmp_seq=11 ttl=64 time=1.022 ms
64 bytes from 10.0.0.100: icmp_seq=12 ttl=64 time=1.032 ms
64 bytes from 10.0.0.100: icmp_seq=13 ttl=64 time=0.776 ms
Request timeout for icmp_seq 14
Request timeout for icmp_seq 15
Request timeout for icmp_seq 16
Request timeout for icmp_seq 17
^C
--- 10.0.0.100 ping statistics ---
19 packets transmitted, 14 packets received, 26.3% packet loss
round-trip min/avg/max/stddev = 0.776/0.996/1.113/0.090 ms
javi@macmini ~ %
```

En esta captura se puede ver como, desde otro equipo de la red hago ping a la nueva IP y confirmo que responde correctamente. Al acabarse el periodo de prueba de netplan la configuración IP vuelve al estado inicial y por tanto pierdo conectividad.

```
sudo netplan apply
```

Finalmente, con el siguiente comando aplico definitivamente la nueva configuración del archivo de configuración. Este cambio será permanente en los siguientes arranques del equipo.

Por si no encuentras los errores en el archivo de configuración anterior, te dejo en esta captura muestra la versión mínima:



```
GNU nano 6.2
# This is the network config written by 'subiquity'
network:
  version: 2
  ethernets:
    enp0s1:
      addresses:
        - 10.0.0.100/24
      nameservers:
        addresses:
          - 8.8.8.8
          - 10.0.0.1
      routes:
        - to: default
          via: 10.0.0.1
```